



"Consult my Web site."

Part 3:

HTML 4.0 Advanced

Forms, Image Maps, and Style Sheets

"I have a dream for the World Wide Web and it has two parts. In the first, the Web becomes a much more powerful means for collaboration between people. I have always imagined the information space as something to which everyone has immediate and intuitive access, and not just to browse, but to create. Furthermore, the dream of people-to-people communication through shared knowledge must be possible for groups of all sizes, interacting electronically with as much ease as they do in person. In the second part of the dream, collaborations extend to computers. Machines become capable of analyzing all the data on the Web—the content, links, and transactions between people and computers."

— Tim Berners-Lee, 1999

Part 3 Description

Welcome to *HTML 4.0 Advanced*, DDC's third part in the *Mastering HTML* series. This course is based on the official HTML 4.01 Specification, released on December 24, 1999 by the World Wide Web Consortium (W3C®). This section is a continuation of DDC's *HTML 4.0 Fundamentals* and *HTML 4.0 Intermediate* courses.

In *HTML 4.0 Advanced*, you will find an in-depth review of forms, image maps, and cascading style sheets. An *HTML 4.0 Advanced Tips & Tricks* Lesson is also included to "fill in the gaps" and to provide students with real-world direction for dealing with scripting problems. The Lesson also suggests techniques for creating impressive effects on Web pages.

Course Objectives

This course was developed for Webmasters, HTML scripters, and anyone publishing Web pages and Web sites. This course utilizes lecture material, hands-on exercises, and lesson-specific quizzes to teach:

- Introduction to forms
- Different form elements
- Client-side image maps
- Embedded fonts
- HTML validators and link checkers
- Web page watermarks
- Cascading style sheets

Follow-up Courses

- DDC's *Web Publishing with Acrobat 4.0*
- DDC's *Mastering JavaScript* Series (2 days)
- DDC's *Mastering VBScript* (2 days)
- DDC's *Creating Web Graphics* (Paint Shop Pro 6 and Photoshop 5.5 versions)
- DDC's *CGI/Perl Fundamentals* (2 days)

Part 3 Setup

DDC's *HTML 4.0 Advanced* requires some PC configuration and setup. Five primary elements are required. Those marked with an "*" are bundled on the Student Files CD-ROM.

Necessary Software

1. Web browser (Microsoft Internet Explorer 5.0 or Netscape Navigator / Communicator 4.0x)
2. Text editor (MS Notepad recommended)
3. Student files*
4. LiveImage image map software*
5. Basic graphics application
 - Paint Shop Pro (www.jasc.com)
 - LView Pro (www.lview.com)
 - Adobe Photoshop (www.adobe.com)



Installing / Launching Software

Three software applications are required to take or teach this course:

- A Web browser (Netscape Navigator 4.0x or higher or Microsoft Internet Explorer (IE) 5.0x or higher).

Navigator can be downloaded from the Netscape Web site at www.netscape.com; Microsoft IE can be downloaded from Microsoft's Web site at www.microsoft.com/ie.

- A text editor (Microsoft Windows Notepad recommended). Notepad can be launched in Windows 98/2000 and Windows NT from the **Start** button on the Task bar (choose **Programs** ► **Accessories** ► **Notepad**).
- LiveImage image map software. The install file, LIVEIMG129INST.EXE, is provided on the Student File CD-ROM in the IMAGE MAP SOFTWARE folder. Note: the free LiveImage evaluation copy is operational for only 14 days from the date of install. You must purchase and register your copy

14

Lesson 14 **Introduction to** **HTML Forms**

Lesson Topics

- ▶ Forms Overview
- ▶ Forms Code Syntax
- ▶ Form Field Types
- ▶ Lesson 14 Summary

Forms Overview

A *form* is not a single HTML element. Rather, it is a collection of data input fields designed to provide an interface by which a Web user can input data, select from among options, or provide personal data. Forms can be simple, complex, short, long, or composed of many different types of fields, as you will learn in this Lesson.

Forms perform two basic roles on the Web:

- gather and upload data from users to an e-mail account
- gather and upload data to a CGI program residing on a Web server²¹

Important for E-commerce

Forms are an increasingly important element in e-commerce Web sites. As business-to-consumer and business-to-business commerce becomes the main function of the Internet's World Wide Web, the commonality and popularity of forms will continue to increase.

An example of a form is shown in Figure 14-1.

The screenshot shows a Netscape browser window titled 'drugstore.com - drugstore.com login - Netscape'. The address bar shows 'store.com/user/login.asp?from=/checkout/place_order.asp&state=state_checkout'. The page content includes a 'Welcome' section with a 'Log on' link, a 'checkout | Create an Account' header, and a section for creating a new account. The form fields are as follows:

- Your e-mail address: johnz@erols.com
- Password of your choice (6 or more characters): [masked]
- Confirm password: [masked]
- Password hint (optional): Mom's favorite color. Used to remind you of your password. Your hint cannot contain your password.
- Title (Mr, Mrs, Mx): Mr.
- First name (How should we greet you?): John
- Last name: Zimbowski

A dashed box labeled 'Form fields' encompasses the input fields for e-mail address, password, confirm password, password hint, title, first name, and last name. At the bottom, there is a checkbox for receiving e-mail from drugstore.com with information about health tips, new site features and special product promotions, which is checked.

Figure 14-1: Example of an HTML form

²¹ For more information regarding CGI and Perl, see DDC's two-day *CGI/Perl Fundamentals* course.

Form Functionality

The functionality of forms is two-fold. In this course, you will learn what is necessary to create a form interface at the user (client) level. But this is only half the equation; the other half occurs at the server level. When a user fills in a form and uploads it, a server takes the submitted data and performs some function.

Name-Value Pairs

Data is transferred from an HTML document at the client level to either a CGI program or e-mail account at the server level in *name-value pairs*, as shown in Figure 14-2.

- **Value:** the variable data being entered by the user (such as their name, address, credit card number, etc.).
- **Name:** the label used to identify the value, which you determined when you create the form.

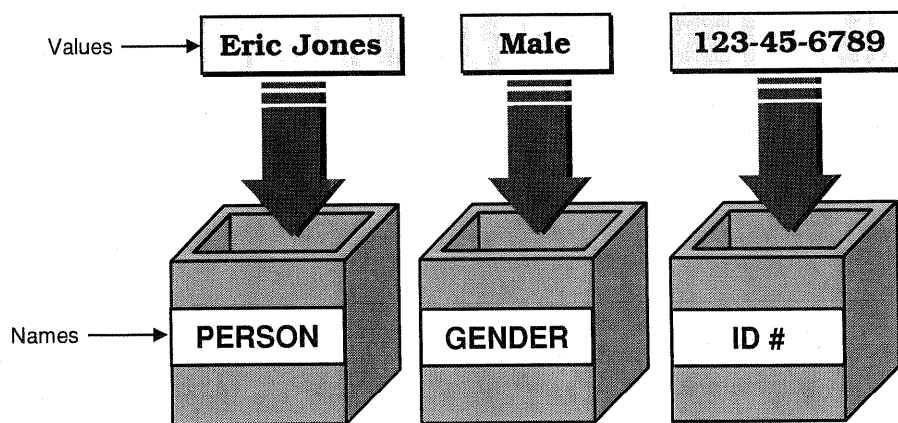


Figure 14-2: Example of name-value pairs in HTML forms

The form field name is simply an identification for the value data which allows the CGI program on a server to separate and differentiate the various values that are being uploaded by the user. This allows the CGI program to “understand” the data it is receiving and properly process it.



Important: *no two values can share the same name.* Each value within a single form must have a unique name. This will become clearer as you progress through this Lesson and create your own forms.

CGI Programs

The server-level programs that receive data submitted from forms (the data submittal process is also called *posting*) and that process it in some manner can be written in one of many programming or scripting languages, including:

- Perl
- C
- C++
- TCL
- AppleScript
- any other Common Gateway Interface (CGI) language²²

CGI programs are most commonly written in Perl. You will learn more about CGI programs in Lesson 15: *Advanced Form Functions*.



You do not necessarily have to write your own CGI programs to handle the output of the forms you create. Many Web hosting vendors provide generic CGI programs to their customers free of charge. There are also many freeware and public domain CGI programs available on the Web.

In a corporate or government agency environment, you may have programmers on staff who are available to help you develop CGI programs that work with and process your forms.

²² See DDC's *CGI/Perl Fundamentals* course for more information on writing CGI/Perl scripts.

Forms Code Syntax

The syntax for an HTML form is similar to that of a table; it is a collection of carefully arranged empty and non-empty tag sets and required attributes.

<FORM> Tag

The <FORM> tag is a non-empty tag (non-empty tags are also called *containers*). One interesting characteristic of forms is that any HTML elements can be placed within the form container, including images, tables, and floating frames. You will nest a table in a form in Lesson 15: *Advanced Form Functions*.

Two Necessary Attributes

The <FORM> tag has two necessary attributes:

- ACTION (<FORM ACTION=mailto:carlp@xyz.com METHOD=post)
- METHOD (<FORM ACTION=mailto:carlp@xyz.com METHOD=post)

ACTION and METHOD are described in more detail in Table 14-1.

<FORM> Tag Attribute	Values
ACTION	<ul style="list-style-type: none"> ▪ The URL of a CGI program that accepts the content of the form uploaded from the client browser. Example: ACTION=/cgi-bin/email.cgi ▪ Also can be an e-mail address using the ACTION=mailto:username@domain.com syntax.
METHOD	<ul style="list-style-type: none"> ▪ <u>get</u>: officially deprecated in the W3C HTML 4.0 Specification; therefore, very uncommon. ▪ <u>post</u>: the default METHOD value. Thus, you will always specify a METHOD value of post, such as METHOD=post.

Table 14-1: ACTION and METHOD: the two necessary <FORM> tag attributes



As you will learn in Lesson 2: *Advanced Form Functions*, you can e-mail the output of a form to multiple e-mail addresses and even specify carbon copies and blind carbon copies.

<INPUT> Tag

<INPUT> is the most common tag within the <FORM> container. <INPUT> specifies the type of form element being created. <INPUT> is used with the following types of form elements (also called *form field types*):

- text fields
- checkboxes
- radio buttons
- password fields

Three Required Attributes

The <INPUT> tag has three required attributes:

- TYPE (<INPUT **TYPE=text** NAME=first_name>)
- NAME (<INPUT TYPE=text **NAME=first_name**>)
- VALUE²³ (<INPUT TYPE=radio NAME=job **VALUE=exec**>)

Required attributes of the <INPUT> tag are described in more detail in Table 14-2.

<INPUT> Attribute	Attribute Values / Description
TYPE	<ul style="list-style-type: none"> ▪ TYPE=text ▪ TYPE=password ▪ TYPE=checkbox ▪ TYPE=radio ▪ TYPE=button
NAME	<ul style="list-style-type: none"> ▪ Any unique identifier of your choice ▪ Same NAME value for all radio buttons or checkboxes in a single group
VALUE	Used only with checkboxes and radio buttons. Each radio button or checkbox must have its own value (but they share the same NAME value).

Table 14-2: <INPUT> tag required attributes

²³ The VALUE attribute is used only with checkboxes and radio buttons.

List Boxes & Text Areas: The Exceptions

There are two major exceptions to the standard rules of form syntax. Neither involves the `<INPUT>` tag; instead, each has its own tag set to form its own container:

- list boxes (also called *selection boxes* or *selections*)
- text areas (similar to text fields to a user, but very different from a script syntax perspective)

NAME = Required Attribute

All form fields have the required attribute of NAME. Despite their unique tag syntax, list boxes and text areas are no exception, as shown in Table 14-3.

Form Field Type	Tag Syntax	Tag Attributes
List Box	<pre><SELECT NAME=bandwidth> <OPTION>33.6 Kbps <OPTION>56.6 Kbps <OPTION>cable modem <OPTION>DSL <OPTION>T-1 <OPTION>T-3 </SELECT></pre>	<ul style="list-style-type: none"> ▪ SIZE: turns a "pop up" list into a scrollable list. SIZE sets the number of list items that are displayed (SIZE=5 would display five list items). ▪ MULTIPLE: allows multiple selections to be made in a single list box (MULTIPLE=5). User selects a contiguous block using <code><SHIFT></code> and non-contiguous list items using <code><CTRL></code>.
Text Area	<pre><TEXTAREA NAME=comments ROWS=5 COLS=65> </TEXTAREA></pre>	<ul style="list-style-type: none"> ▪ ROWS: necessary attribute that defines the height of the text area field, as defined in text rows. ROWS=6 would display six rows of text. A user can input more than six rows, but additional text must be scrolled to be viewed. ▪ COLS: necessary attribute that defines the width of the text area field, as defined in characters. COLS=65 would display a text area that is 65 characters wide.

Table 14-3: List box and text area characteristics

Putting It All Together

As you can see, one HTML form can be very different from another. You may need to create a very basic form with only a couple of text fields, or you may need to create dozens of fields and use each type of field multiple times.

The code for a prototypical form involving each type of form field is displayed below. The corresponding interpretation by a browser is shown in Figure 14-3 on the following page.

```
<H3>Open a New Account with Us</H3>
<FORM ACTION=mailto:webmaster@corp.com ENCTYPE=text/plain METHOD=post>
First Name: <INPUT TYPE=text NAME=first_name><BR>
Last Name: <INPUT TYPE=text NAME=last_name><P>
Address: <INPUT TYPE=text NAME=address><BR>
City: <INPUT TYPE=text NAME=address><BR>
State: <SELECT NAME=state>
      <OPTION>Alabama
      <OPTION>Alaska
      <OPTION>Arizona
      <OPTION>Arkansas
      <OPTION>California
      <OPTION>Colorado
    </SELECT><BR>

<!-- this list should obviously contain 50 <OPTION> tags, one for each
state; for this example, only the first six states are used -->

ZIP Code: <INPUT TYPE=text NAME=zip><P>
Credit Card:<BR>
<INPUT TYPE=radio NAME=cardtype VALUE=Visa> Visa<BR>
<INPUT TYPE=radio NAME=cardtype VALUE=Mastercard> Mastercard<BR>
<INPUT TYPE=radio NAME=cardtype VALUE=Amex> American Express<P>

Credit Card #: <INPUT TYPE=password NAME=cardnum><P>
Your Hobbies:<BR>
<INPUT TYPE=checkbox NAME=hobby VALUE=running> Running<BR>
<INPUT TYPE=checkbox NAME=hobby VALUE=stamps> Stamp Collecting<BR>
<INPUT TYPE=checkbox NAME=hobby VALUE=chess> Chess<BR>
<INPUT TYPE=checkbox NAME=hobby VALUE=twister> Twister<P>
We welcome your comments and feedback:<BR>
<TEXTAREA ROWS=4 COLS=65> </TEXTAREA><P>

<INPUT TYPE=submit> <INPUT TYPE=reset>

</FORM>
```

The browser interpretation of the form code on the previous page is shown in Figure 14-3.

Prototypical Form - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address D:\Courseware\HTML 4.0 Advanced\form-prototype.htm

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit

Open a New Account with Us

First Name:

Last Name:

Address:

City:

State:

ZIP Code:

Credit Card:

☐ Visa

☐ Mastercard

☒ American Express

Credit Card #:

Your Hobbies:

☒ Running

☐ Stamp Collecting

☐ Chess

☒ Twister

We welcome your comments and feedback:

Done My Computer

Figure 14-3: HTML form code interpreted by a Web browser (IE 5)



In the code on the previous page, the `<OPTION>` tags do not have `VALUE` attributes. While `VALUE` attributes are not prohibited and are part of the W3C HTML 4.0 Specification, both MS Internet Explorer and Netscape Navigator/Communicator do not require them.

Form Field Types

Forms are comprised of fields. There are six basic types of fields. Each field type is designed to capture a particular type of data or to capture data in a particular manner.

Table 14-4 lists all six form field types and provides a basic description. Note that each form field is created using the `<INPUT>` tag with a different `TYPE` attribute, with the exception of list boxes and text areas.

Form Field Type	Function	HTML Tag Syntax
Checkbox	Offer non-exclusive selection choices to a user. In other words, multiple options are presented to the user and they can choose none, one, or all of the options.	<code><INPUT TYPE=checkbox NAME=sports VALUE=baseball></code>
List Box	Offer multiple options to a user in a vertical list; the user chooses a single answer. More technically, this is called a <i>select list</i> .	<code><SELECT NAME=state> <OPTION>Alabama <OPTION>Alaska <OPTION>Arizona <OPTION>Arkansas </SELECT></code>
Password	Identical to a text field, except all data input into the field is displayed in asterisks for security purposes. If you are collecting sensitive data, such as social security number, credit card number, bank account number, etc., you should use a password field.	<code><INPUT TYPE=password NAME=creditcard></code>
Radio Button	Offer exclusive options to a user. In other words, multiple options are presented to the user, only <i>one</i> of which can be chosen.	<code><INPUT TYPE=radio NAME=agegroup VALUE=teen></code>
Text	The most basic type of form field that allows a user to input any alphanumeric data in a single line of a specific length (number of characters). You can limit the amount of text that can be typed in the text field and also determine the width of the text field as displayed in the user's browser. Similar to a text area field, but designed to accommodate much smaller text blocks.	<code><INPUT TYPE=text NAME=first_name></code>
Text Area	A text field designed to accommodate a relatively large amount of alphanumeric information on multiple lines. Like a text field, you can define the size of a text area.	<code><TEXTAREA NAME=comments ROWS=5 COLS=65> </TEXTAREA></code>

Table 14-4: Types of HTML form fields

Text Fields

Text fields are the most common type of form fields and also the easiest to script. Text fields are typically used to request brief answers or personal data from a user, such as their name or address.

Characteristics

- Any alphanumeric data can be entered into a text field.
- The width of a text field can be specified using the `SIZE` attribute; the value is measured in characters (example: `SIZE=30`).
- The maximum number of characters that can be input into a text field—regardless of its width (as determined by the `SIZE` attribute)—can be specified using the `MAXLENGTH` attribute (example: `MAXLENGTH=25`).
- Most browsers will display a text field as 20 characters wide if you do not use the `SIZE` attribute.

Example

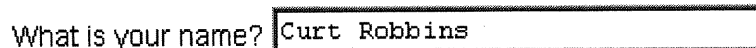
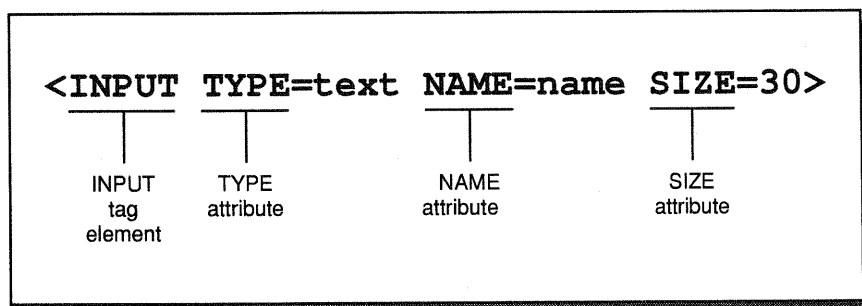


Figure 14-4: Example of a text field

Code

What is your name?



Password Fields

Password fields are very similar to text fields. The difference is that password fields do not display what a user types into them. Instead, a password field displays only asterisks. This serves the purpose of security. If a user is inputting her social security number or a credit card number into a form field, anyone looking over her shoulder will see only asterisks.

Characteristics

- Any alphanumeric data can be entered into a password field.
- The width of a text field can be specified using the `SIZE` attribute; the value is measured in characters (example: `SIZE=30`).
- The maximum number of characters that can be input into a text field—regardless of its width (as determined by the `SIZE` attribute)—can be specified using the `MAXLENGTH` attribute (example: `MAXLENGTH=25`).
- This is visual security; no encryption whatsoever is used when the data is uploaded (unless a security scheme, such as SSL, is being employed).

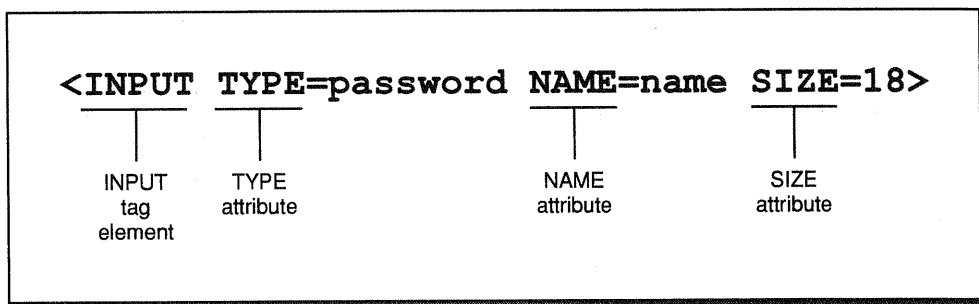
Example

What is your credit card number?

Figure 14-5: Example of a password field

Code

What is your credit card number? `<INPUT TYPE=password NAME=name SIZE=18>`



Checkboxes

Checkboxes are used to capture non-exclusive data. *What?* In plain English, checkboxes allow a user to choose from multiple options. The user can choose any number of the available checkbox options (none, one, all, etc.). Thus, checkboxes involve a non-exclusive logic, where no single option is exclusive.

Characteristics

- Checkboxes are only identified to users according to the text that is adjacent to each checkbox. Thus, you must script the data in plain body text that identifies each checkbox for the user.
- Checkboxes require an additional `<INPUT>` tag attribute: `VALUE`. All checkboxes in a single group have the same `NAME` value, but each checkbox is individually identified within the form using the `VALUE` attribute.
- Only checkbox and radio button form field types require the `VALUE` attribute.
- Each checkbox can be selected (checked) or unselected (not checked).

Example

Favorite Exercise
☒ Running
☒ Bicycling
☐ Aerobics
☒ Swimming
☐ Watching TV

Figure 14-6: Example of checkboxes

Code

```
Favorite Exercise<BR>
<INPUT TYPE=checkbox NAME=sports VALUE=run> Running<BR>
<INPUT TYPE=checkbox NAME=sports VALUE=cycle> Bicycling<BR>
<INPUT TYPE=checkbox NAME=sports VALUE=aerobics> Aerobics<BR>
<INPUT TYPE=checkbox NAME=sports VALUE=swim> Swimming<BR>
<INPUT TYPE=checkbox NAME=sports VALUE=tv> Watching TV<BR>
```

Radio Buttons

Radio buttons are very similar to checkboxes. Visually, radio buttons appear as circles (whereas checkboxes are displayed as squares). Logically, radio buttons are exclusive. If you create a group of radio buttons and present six options, the user can select only one answer.

Characteristics

- Radio buttons are only identified to users according to the text that is adjacent to each radio button. Thus, you must script the data in plain body text that identifies each radio button for the user.
- Radio buttons require an additional `<INPUT>` tag attribute: `VALUE`. All radio buttons in a single group have the same `NAME` value, but each radio button is individually identified within the form via the `VALUE` attribute.
- You can use an unlimited number of radio buttons in a single group or in a form.
- Only one radio button in a single group can be selected.

Example

What are you?
☐ Infant
☐ Child
☐ Teenager
☒ Adult
☐ Senior Citizen
☐ Dog

Figure 14-7: Example of radio buttons

Code

```
What are you?<P>
<INPUT TYPE=radio NAME=name VALUE=infant> Infant<BR>
<INPUT TYPE=radio NAME=name VALUE=child> Child<BR>
<INPUT TYPE=radio NAME=name VALUE=teenager> Teenager<BR>
<INPUT TYPE=radio NAME=name VALUE=adult> Adult<BR>
<INPUT TYPE=radio NAME=name VALUE=senior> Senior Citizen<BR>
<INPUT TYPE=radio NAME=name VALUE=arf> Dog<BR>
```

Text Areas

A text area field is similar in appearance and function to a text field, but the HTML code behind it is very different. Whereas a text field is designed for a single line of text, a text area is designed for a much larger volume of text, such as a paragraph or even several paragraphs. You determine how big and how much text a text area field will accommodate.

Characteristics

- A text area is not sized with a `SIZE` attribute; rather, it is sized using `ROWS` and `COLS` attributes (similar to HTML tables).
- A text area is a container; in other words, it involves an opening and closing `<TEXTAREA>` tag set (unlike the empty `<INPUT>` tag for text, password, checkbox, and radio button fields).
- Vertical and horizontal scroll bars automatically appear; they are greyed out until a user types more information than is visible in the text area.
- Text typed into a text area field automatically word-wraps in Internet Explorer, but does not in Netscape Navigator (in which it bleeds off the field).

Example

We welcome your feedback!

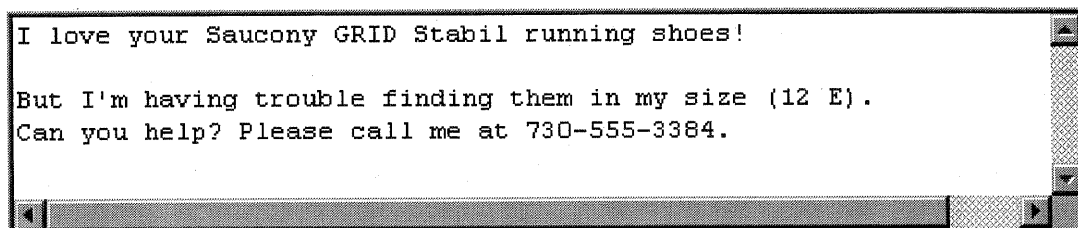


Figure 14-8: Example of a text area field

Code

```
We welcome your feedback!<P>
<TEXTAREA NAME=comments ROWS=5 COLS=65> </TEXTAREA>
```

List Boxes

A list box, also called a *selection list*, *selection box*, or simply a *menu*, is identical in logic to radio buttons; it presents a list of options and only one option can be chosen by the user. The difference—other than appearance—between list boxes and radio buttons is that list boxes are designed to provide many more options without taking more screen space.

A list box allows you to present a user with dozens or even hundreds of options in the same space as a basic text field; something not practical using radio buttons.

Characteristics

- The select tag is a container (non-empty tag set); thus, you must use both the opening and closing <SELECT> tags to create a list box.
- Similar to an ordered or unordered list in HTML, you must use <OPTION> tags within the <SELECT> tags in order to create a list box.

Example

What is your Internet connection bandwidth?


asymmetrical cable modem 

Figure 14-9: Example of a list box

Code

```
What is your Internet connection bandwidth?<P>
<SELECT name=bandwidth>
<OPTION>28.8 Kbps dialup modem
<OPTION>33.6 Kbps dialup modem
<OPTION>56.6 Kbps dialup modem
<OPTION>64 Kbps ISDN
<OPTION>128 Kbps ISDN
<OPTION SELECTED>asymmetrical cable modem
<OPTION>symmetrical cable modem
<OPTION>cable modem (unsure of type)
<OPTION>DSL
<OPTION>T-1
<OPTION>T-3
<OPTION>None of your business!
</SELECT>
```

Submit & Reset Buttons

Regardless of the simplicity or complexity of your form, a mechanism must exist that allows a user to upload data to an e-mail address or CGI program on a server.

This is the role of the *Submit* and *Reset* buttons. Created by using the now-familiar `<INPUT>` tag, the Submit and Reset buttons are easy and straightforward to create. By default, the faces of the Submit and Reset buttons read “Submit Query” and “Reset,” respectively. This can be changed, as you will learn in Lesson 15: *Advanced Form Functions*.

Characteristics

- The Submit button executes the actions specified by the values of the attributes in the opening `<FORM>` tag.

Thus, assume your opening `<FORM>` tag appeared as follows:

```
<FORM ACTION=mailto:webmaster@company.com METHOD=post>
```

The contents of the form fields would be uploaded via e-mail to the address `webmaster@company.com`. If the `ACTION` value was the URL of a CGI program, the form field contents would be uploaded to the CGI program.

- The Reset button is very simple in function; it simply clears the contents of the form fields. Thus, a user *populates* the form fields and the Reset button *de-populates* them.
- Unlike most other form fields involving the `<INPUT>` tag, Submit and Reset buttons do not take the `NAME` attribute.

Example

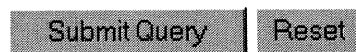


Figure 14-10: Example of Submit and Reset buttons

Code

```
<INPUT TYPE=submit> <INPUT TYPE=reset>
```

Exercise 14-1: Creating a Basic Form

In this exercise, you will script a form containing a text field, a password field, and a selection of checkboxes.

1. Launch your Web browser (Netscape Navigator/Communicator or Microsoft Internet Explorer).
2. Launch Microsoft Notepad (or the plain text editor of your choice).
3. Using Notepad, open FIRST-FORM.HTM in the HTML-3 folder on your Desktop.
4. Replace "STEP 1" with the following code (substitute the e-mail address of the computer on which you are working for the text "your e-mail address"):

```
<FORM ACTION=mailto:your e-mail address ENCTYPE=text/plain
METHOD=post>
```



ENCTYPE formats the output of a form to make it more readable in an e-mail client application. You will learn more about ENCTYPE in Lesson 2.

5. Replace "STEP 2" with the following code:

```
<INPUT TYPE=text NAME=name>
```

6. Replace "STEP 3" with the following code:

```
<INPUT TYPE=checkbox NAME=sport VALUE=run> Running<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=cycle> Bicycling<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=aerobics> Aerobics<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=swim> Swimming<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=TV> Watching TV<P>
```

7. Replace "STEP 4" with the following code:

```
<INPUT TYPE=radio NAME=whatru VALUE=infant>Infant<BR>
<INPUT TYPE=radio NAME=whatru VALUE=child>Child<BR>
<INPUT TYPE=radio NAME=whatru VALUE=teenager>Teenager<BR>
<INPUT TYPE=radio NAME=whatru VALUE=adult>Adult<BR>
<INPUT TYPE=radio NAME=whatru VALUE=senior>Senior Citizen<BR>
<INPUT TYPE=radio NAME=whatru VALUE=arf>Dog<P>
```

8. Save your changes to FIRST-FORM.HTM.
9. Toggle over to your Web browser (<ALT + TAB>).

10. Open FIRST-FORM.HTM in your Web browser:

- ▶ In IE 5, select <CTRL + O>, click **Browse**, open the HTML-3 folder on your Desktop, and double-click FIRST-FORM.HTM.

or

- ▶ In Navigator, select <CTRL + O>, click **Choose File**, open the HTML-3 folder on your Desktop, and double-click FIRST-FORM.HTM.



Your browser should display the page, as shown in Figure 14-11.

The screenshot shows a Microsoft Internet Explorer window titled "Basic Form Field - Microsoft Internet Explorer". The address bar shows the path "C:\WINNT\Profiles\Administrator\Desktop\HTML-3\first-form.htm". The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The toolbar contains buttons for Back, Forward, Stop, Refresh, Home, Search, Favorites, History, Mail, Print, and Edit. The main content area displays a web form with the following sections:

- Basic Form Fields**
- Basic Text Field**: A text input field with the label "What is your name?"
- Checkboxes**: A section titled "Favorite Form of Exercise" with five checkboxes: Running, Bicycling, Aerobics, Swimming, and Watching TV.
- Radio Buttons**: A section titled "What are you?" with six radio buttons: Infant, Child, Teenager, Adult, Senior Citizen, and Dog.

The status bar at the bottom shows "Done" and "My Computer".

Figure 14-11: Form featuring text, checkboxes, and radio button fields

Exercise 14-2: Creating Password Fields, Text Areas, and List Boxes

In this exercise, you will enhance the form you created in the previous exercise by adding a password field, a text area, and a list box. You will also add Submit and Reset buttons to enable users to upload the contents of the form fields to your e-mail account.

1. Be sure FIRST-FORM.HTM is open in both your text editor and your Web browser.
2. In your text editor, replace “STEP 5” with the following code:

```
<INPUT TYPE=password NAME=cardnum>
```

3. Replace “STEP 6” with the following code:

```
<TEXTAREA NAME=feedback ROWS=5 COLS=65> </TEXTAREA>
```

4. Replace “STEP 7” with the following code:

```
<SELECT NAME=bandwidth>
<OPTION>28.8 Kbps dialup modem
<OPTION SELECTED>56.6 Kbps dialup modem
<OPTION>ISDN 64 Kbps
<OPTION>ISDN 128 Kbps
<OPTION>asymmetrical cable modem
<OPTION>symmetrical cable modem
<OPTION>DSL
<OPTION>satellite
<OPTION>T-1
<OPTION>T-3
</SELECT>
```

```
<HR WIDTH=15% ALIGN=left SIZE=5><P>
```



Inserting the `SELECTED` attribute in a particular `<OPTION>` tag will display that option when a user views the form.

5. You now need to create the Submit and Reset buttons to enable users to upload form field contents or clear all form fields. Replace “STEP 8” with the following code:

```
<INPUT TYPE=submit> <INPUT TYPE=reset>
```

6. Replace “STEP 9” with the following code:

```
</FORM>
```

7. Save FIRST-FORM.HTM.

8. Toggle over to your Web browser.
9. Reload the Web page (<CTRL + R>).



The form fields you added in the preceding steps of this exercise should appear, as shown in Figure 14-12.

The screenshot shows a Microsoft Internet Explorer window titled "Basic Form Field - Microsoft Internet Explorer". The address bar shows the file path "D:\Courseware\HTML 4.0 Advanced\Student Files\first-form solution.htm". The form contains the following elements:

- Password Field:** A text input field with the label "What is your credit card number?". An annotation points to it with the text "Asterisks will appear in password field".
- Text Area:** A large text input area with the label "We welcome your feedback!". An annotation points to it with the text "As scripted, an unlimited amount of text can be typed in the text area".
- List Box:** A dropdown menu with the label "What is your Internet connection bandwidth?". The selected option is "56.6 Kbps dialup modem".
- Submit and Reset buttons:** Two buttons labeled "Submit Query" and "Reset". An annotation points to them with the text "Submit and Reset buttons".

The browser's status bar at the bottom shows "Done" and "My Computer".

Figure 14-12: Completed form, including Submit and Reset buttons at bottom

Exercise 14-3: Testing Your Form

In this exercise, you will test your form by populating each field and uploading it. You should always thoroughly test your form to ensure that it functions properly. You want to identify and repair any errors before your form goes live online. Allowing users to identify and inform you of errors is unprofessional.

1. In your Web browser, open FIRST-FORM.HTM (if necessary).
2. At the top of the form, in the basic text field, type your name.
3. Press <TAB> to advance to the next form field. The first checkbox will appear with a marquee around it, denoting that it is the active field.
4. On your keyboard, press the space bar. The Running checkbox will be selected.
5. Press <TAB> three times to activate the Swimming checkbox. Press the spacebar.
6. Scroll down to the radio buttons section and click **Adult**.
7. Click **Senior Citizen**.



The Senior Citizen radio button will be selected while the Adult radio button will be deselected. Remember: radio buttons are exclusive; only one in a single group can be selected.

8. In the password field, type any 15-digit number. If you coded the password <INPUT> tag correctly, nobody around you should be able to read the numbers you type.
9. In the text area field, type the following text: **Nice form. But why am I giving you my credit card number? Is this legal?!**
10. In the list box, select **symmetrical cable modem**.
11. At the bottom of the field, click the **Submit Query** button.



The contents of the form fields will be uploaded via e-mail to your e-mail account.

If an alert box appears asking you to save passwords or warning you that the data you are uploading is not encrypted, do not save passwords and choose whatever option continues the upload of your form data.

12. If you have an e-mail account and e-mail software on the computer on which you are working, launch your e-mail software and download all new messages.



You should have an incoming e-mail message that appears similar to Figure 14-13. Note: the heading information (date, from, subject, etc.) will vary from the example shown below).

Data
submitted by
user and
uploaded by
the form

Curt Robbins, 02:12 PM 1/16/00 -0800, Form posted from Mozilla

Subject: Form posted from Mozilla

Date: Sun, 16 Jan 2000 14:12:11 -0800
 From: Curt Robbins <toshi@quessing.com>
 Subject: Form posted from Mozilla
 To: webmaster@quessing.com
 Organization: Quessing Courseware Corp.
 X-Mailer: Mozilla 4.7 [en] (Win98; I)
 X-Accept-Language: en
 X-Loop-Detect: 1

name=Curt Robbins
 activities=running
 activities=swimming
 whatru=senior
 creditcard=453849983090238
 comments= Nice form. But why am I giving you my credit card number? Is this legal?!
 bandwidth=symmetrical cable modem

Figure 14-13: Form data as uploaded to an e-mail account



List boxes are most useful when there are many choices to be made. If there are only a few choices, then either checkboxes or radio buttons are typically a more logical choice, providing a more streamlined, intuitive interface for users.

Exercise 14-4: Going Solo—Scripting an HTML Form on Your Own

In this exercise, you will create an HTML form on your own that contains text fields, a password field, radio buttons, checkboxes, list boxes, and a text area.

1. Using the HTML form knowledge you have gained from previous exercises, script an HTML document that will result in the form shown in Figure 14-14. Give the page an HTML title of “My First Solo Form.”

My First Solo Form - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address Links

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit

Acme Products, Inc. Survey

Thanks for participating in the Acme Products, Inc. Survey. Each month, we draw two lucky names from our virtual fishbowl. Each lucky winner will receive dinner for two at Friday's and a pair of Acme T-shirts!

First Name:

Last Name:

Gender: ☐ Male ☒ Female

Age:

City:

State:

Of the following activities, which is your favorite?

☒ Running

☐ Chess

☐ Wood Working

☐ Sewing

☐ Camping

☐ Bar Fights

What is your Social Security Number?

What is the craziest thing that ever happened to you when camping?

Which of the following are true regarding you?

☒ I feel healthy

☐ I am creative

☒ I love pets

☐ I hate *Touched by an Angel*

☐ I love sports cars

☒ I drink too much

☐ I am religious

Done My Computer

Figure 14-14: Your first challenge—create this form

2. Make the **ACTION** of the form to send an e-mail message to your personal e-mail account.
3. Make the **Age** field a list box with the following **OPTIONS** values: 18-25, 26-33, 34-41, 42-49, 50-57, 58-65, 66-73, 74-81, 82-89, 90-97.
4. For the **State** list box, include all 50 states in the U.S. Do not use abbreviations.



To save you time, you can open the file STATES.TXT in the HTML-3 folder on your Desktop.

Hint: copy the text from the file STATES.TXT into the Windows clipboard (<CTRL + C>) and paste it into your form. Then script one <OPTION> tag left adjacent of Alabama. Copy this tag to the clipboard and paste it 49 times left adjacent of each state name.

5. When you are finished, save the file as SOLO FORM.HTM and save it to the HTML-3 folder on your Desktop.
6. Open and view the completed page in your Web browser. Compare it to Figure 14-14 on the previous page. How well do they compare?
7. There will probably be minor errors. Return to your text editor and fix these problems. Check the document in your Web browser to ensure that the problems have been fixed properly.



If you are participating in an instructor-led environment and get confused or have trouble, ask your instructor for assistance.

Lesson 14 Summary

- ▶ A form is not a single HTML element. Rather, it is a collection of data input fields designed to provide an interface by which a Web user can input data, selecting from among options or providing personal information. Forms perform two basic roles on the Web: 1) gather and upload data from users to an e-mail account and 2) gather and upload data to a CGI program residing on a Web server.
- ▶ Data is transferred from an HTML document at the client level to either a CGI program or e-mail account at the server level in name-value pairs. The “value” is the variable data being entered by the user; the “name” is used to identify the value.
- ▶ Forms have a relatively sophisticated and strict syntax that is composed of a family of tags, containers (tag sets), and attributes (many of which are necessary). The overriding container is formed by the opening and closing `<FORM>` tags. `<FORM>` has two necessary attributes: `ACTION` and `METHOD`.
- ▶ The most common tag inside the `<FORM>` container is the `<INPUT>` tag. `<INPUT>` determines the type of form field being created. All types of form fields are created using the `<INPUT>` tag, with the exception of list boxes and text areas.
- ▶ Text fields are the most common type of form field. HTML syntax is: `<INPUT TYPE=text NAME=first_name>`.
- ▶ Password fields are similar to text fields. The only difference is that the text entered by a user in a password field is displayed as asterisks. This is because password fields are used to collect confidential information, such as credit card numbers and social security numbers. HTML syntax is: `<INPUT TYPE=password NAME=ssn>`.
- ▶ Checkboxes capture *non-exclusive* data. In other words, a user can choose any number of checkboxes within a single group (none, any, or all). HTML syntax is: `<INPUT TYPE=checkbox NAME=fave_sports VALUE=football>`. Checkboxes, like radio buttons, have an additional necessary attribute of `VALUE`.
- ▶ Radio buttons capture *exclusive* data. Thus, they provide options to a user where only one answer from among the options is logical. Examples are: age group, income level, race, etc. HTML syntax is: `<INPUT TYPE=radio NAME=job_type VALUE=exec>`.
- ▶ Text areas are like text fields on steroids. They have multiple lines and can be as wide as the page. They are designed to accept a large amount of alphanumeric data. The HTML syntax is: `<TEXTAREA NAME=comments ROWS=5 COLS=65> </TEXTAREA>`.
- ▶ List boxes are similar in logic to radio buttons in that they provide a list of options, only one of which can be selected. The HTML syntax is: `<SELECT NAME=state>
<OPTION>item 1<OPTION>item 2<OPTION>item 3<OPTION>item 4</SELECT>`.
- ▶ The data populating form fields is uploaded to a CGI program or e-mail account by the user via the Submit button. There is also a Reset button, which simply clears all fields in a form. HTML syntax is: `<INPUT TYPE=submit>` and `<INPUT TYPE=reset>`.